Extraction of numerical residues in families of levelings

Wonder Alexandre Luz Alves, Alexandre Morimitsu, Joel Sánchez Castro and Ronaldo Fumio Hashimoto
Department of Computer Science, Institute of Mathematics and Statistics,
University of São Paulo, São Paulo, Brazil
wonder@ime.usp.br

***Abstract*—This work introduces a residual operator called ultimate attribute leveling. We also present an efficient algorithm for ultimate attribute leveling computation by using a structure called tree of shapes. Our algorithm for computating ultimate attribute leveling is based on the fact (proved in this work) that levelings can be obtained by pruning nodes from the tree of shapes. This is a novel result, since so far it is known that levelings can be obtained from component trees. Finally, we propose the use of ultimate attribute leveling with shape information to extract contrast using a priori knowledge of an application. Experimental results applied to text location show the potentiality of using ultimate attribute leveling with shape information for solving problems in image processing area.**

***Keywords*-residual morphological operator; ultimate attribute opening; ultimate attribute closing; ultimate attribute leveling.**

I. Introduction

Mathematical Morphology (MM) is a theory that studies mappings between complete lattices [1], [2]. In particular, mappings on the set of all gray level images $\mathcal{F}_{\mathcal{D}}$ defined on domain $\mathcal{D} \subset \mathbb{Z}^2$ are of special interest in MM and they are called *operators*. Given two operators, say ψ and φ, the operator obtained by the difference between them, that is, $r = \psi - \varphi$, is known as *residual operator*. In the literature, it is possible to find many applications of residual operators in Image Processing: morphological gradients (which are the residue of a dilation and an erosion, i.e., $\delta - \varepsilon$), white top hats (which are the residue of the identity operator and an opening, i.e., $\iota - \gamma$); and black top hats (which are the residue of a closing and the identity operator, i.e., $\phi - \iota$) are examples of residual operators.

The residual operator can be extended to a family of operators $\{\psi_i\}$ and $\{\varphi_i\}$ as the supremum of residual operators $\{r_i : r_i = \psi_i - \varphi_i\}$, i.e., $r = \sup_i\{r_i\}$ [3]. For example, the *ultimate opening* is defined as the supremum of residual operators when ψ_i and φ_i are consecutive openings, i.e., $r_i = \gamma_i - \gamma_{i+1}$ [3]. These residual operators have been extended to extract residues of families of connected operators [4], [5], [6]. At first, Retornaz and Marcotegui [6] proposed the *ultimate attribute opening* for scene text localization. Then, Fabrizio and Marcotegui [4] developed an efficient algorithm for computing ultimate attribute opening using component trees [7]. Later, Hernández and Marcotegui [8] added shape information to extract residues from ultimate atribute opening.

Differently from the works [4], [6], [8], Meyer [5] proposed to extract residues from a family of *levelings* obtained by using erosions and dilations as markers. The obtained levelings produce positive and negative residues (which is not the case for the ultimate opening). In this way, Meyer [5], in contrast to Beucher [3], treats separately the positive and negative residues. In this work, we call this residual operator as *ultimate leveling*.

Given the above considerations, in this work, we provide several original results: (i) we prove that we can obtain levelings from tree of shapes [9] and we call them attribute leveling (it is already known in the literature that we can extract levelings from component trees); (ii) we propose an efficient algorithm for computing ultimate attribute leveling which is an extension of the algorithm for computing ultimate attribute opening presented by Fabrizio and Marcotegui [4] (differently to latter work, our algorithm can also use tree of shapes to compute ultimate attribute leveling); (iii) we added shape information to the ultimate attribute leveling (differently to the work of Hernández and Marcotegui [8] which added shape information to ultimate attribute opening).

The remainder of this paper is structured as follows. For the sake of completeness, Sections II, III and IV briefly recall some definitions and properties of image representation by tree structures, connected filters, and constrast extraction using residues, respectively. In particular, Section III provides the first original result of this work. The other original contribution of this paper is given in Section V, where we introduce the ultimate attribute leveling and present an efficient algorithm for its computation. In Section VI, we present the third contribuition of this work: the ultimate attribute leveling with shape information. Experimental results are shown in Section VII, and finally, Section VIII concludes this work and presents some future research direction.

II. Image representation using tree structures

There are many different ways of building trees that represent a given image, such as component trees [7], tree of shapes [9] and binary partition trees [10]. Each of these structures is able to fully reconstruct the image using its hierarchy and information stored in the nodes of the tree, which varies according to the chosen tree.

The main advantage of using such representations is that the hierarchy of the tree is directly related to some kind of

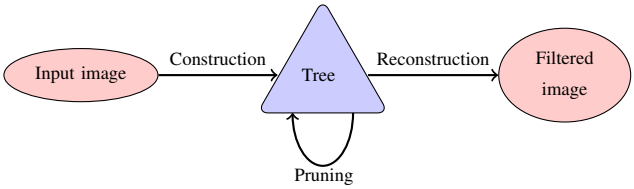

Figure 1. Image representation by tree structures.

relation between the regions of the image. Modifications in the nodes of the trees will imply modifications of regions once the image is reconstructed from the modified tree. This allows, for example, an efficient computation of regions simplifications of the original image since this can be done by simply pruning nodes of the tree and reconstructing the image from the pruned tree (see Fig. 1).

A. Component Trees

Given an image $f : \mathcal{D}_f \subset \mathbb{Z}^2 \to \{0, 1, ..., 255\}$, we define $\mathcal{X}^\mu(f) = \{p \in D_f : f(p) \leq \mu\}$ as the *upper level set at value* μ. Likewise, the set $\mathcal{X}_\lambda(f) = \{p \in \mathcal{D}_f : f(p) \geq \lambda\}$ will be defined as the *lower level set at value* λ. These level sets are nested and, consequently, the family of upper (resp. lower) sets is increasing (resp. decreasing). Therefore, $\forall \alpha, \beta \in \{0, 1, \ldots, 255\}$, such that $\alpha < \beta$, $\mathcal{X}_\alpha(f) \subseteq \mathcal{X}_\beta(f)$ and $\mathcal{X}^\alpha(f) \supseteq \mathcal{X}^\beta(f)$. The image f can be reconstructed using either the family of lower or upper sets [9]. In fact, $\forall x \in \mathcal{D}_f$,

$$f(x) = \sup\{\lambda : x \in \mathcal{X}_\lambda(f)\} = \inf\{\mu : x \in \mathcal{X}^\mu(f)\}.$$

Let $\mathcal{L}(f)$ be the family of lower connected components (CCs) of level sets (i.e. $\mathcal{L}(f) = \{C : C$ is a CC of $\mathcal{X}_\lambda(f), 0 \leq \lambda \leq 255\}$) and let $\mathcal{U}(f)$ be the family of CCs of upper level sets (i.e. $\mathcal{U}(f) = \{C : C$ is a CC of $\mathcal{X}^\mu(f), 0 \leq \mu \leq 255\}$). Then, the pair ordered consisting of the family of CCs of lower (resp., upper) level sets and the usual set inclusion relation $(\mathcal{L}(f), \subseteq)$ (resp. $(\mathcal{U}(f), \subseteq)$) induces a tree structure. This leads us to Def. 1, and consequently to Prop. 1.

Definition 1. *Let $(\mathcal{T}, \preceq)$ be an ordered set. We say that $\preceq$ induces a tree structure in $\mathcal{T}$ if the two conditions hold:*

1) *$\exists R \in \mathcal{T}$ such that $\forall N \in \mathcal{T}, N \preceq R$. In that case we shall say that R is root of tree.*
2) *$\forall A, B, C \in \mathcal{T}$, if $A \preceq B$ and $A \preceq C$ then either $B \preceq C$, or $C \preceq B$. In that case, we shall say that B and C are nested.*

Proposition 1. *Both $(\mathcal{L}(f), \subseteq)$ and $(\mathcal{U}(f), \subseteq)$ are trees.*

Thus, its nodes $C_{\lambda,k}$ (resp., $C^{\mu,k}$) represent the k-th CCs of $\mathcal{X}_\lambda(f)$ (resp. $\mathcal{X}^\mu(f)$) and two nodes $C_{\lambda,k}$ and $C_{\lambda',k'}$ (resp., $C^{\mu,k}$ and $C^{\mu',k'}$) are linked by an edge if, and only if, $\nexists C_{\lambda'',k''} \in \mathcal{L}(f)$ (resp., $\nexists C^{\mu'',k''} \in \mathcal{U}(f)$) such that either $C_{\lambda,k} \subseteq C_{\lambda'',k''} \subseteq C_{\lambda',k'}$ (resp., $C^{\mu,k} \subseteq C^{\mu'',k''} \subseteq C^{\mu',k'}$), or $C_{\lambda',k'} \subseteq C_{\lambda'',k''} \subseteq C_{\lambda,k}$ (resp., $C^{\mu',k'} \subseteq C^{\mu'',k''} \subseteq C^{\mu,k}$).

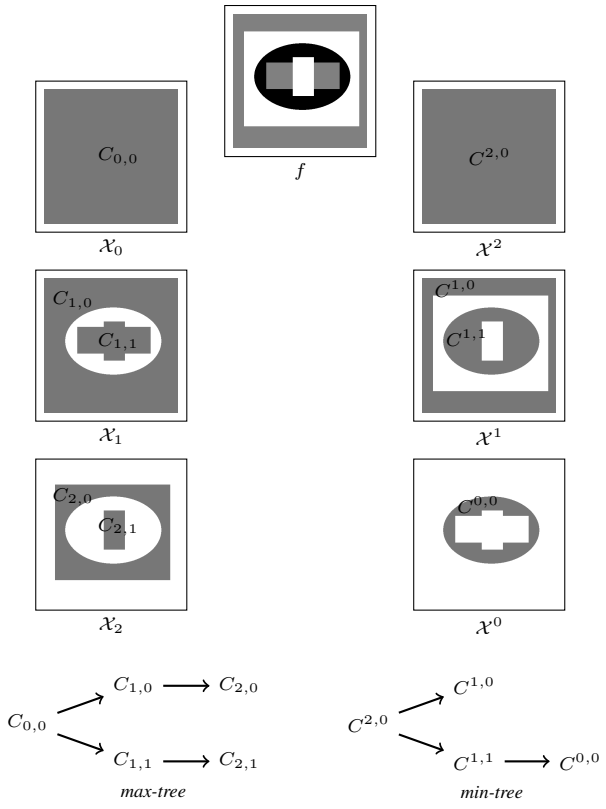

Figure 2. Example of component trees. At the top-center is the input image f; the left and right columns are, respectively, the CCs of the levels sets $\mathcal{X}_\lambda$ and $\mathcal{X}^\mu$; and at the bottom are the trees: max-tree and min-tree.

In this representation, regional maxima (resp. minima) of the image f are represented by leaf nodes and for this reason this tree is also known as *max-tree* (resp. *min-tree*). Fig. 2 shows an image f, their respectives level sets $\mathcal{X}_\lambda(f)$ and $\mathcal{X}^\mu(f)$, the CCs $C_{i,j} \in \mathcal{L}(f)$ and $C^{i,j} \in \mathcal{U}(f)$ and the trees. The algorithms for the component trees construction can be found in [4], [7].

B. Tree of shapes

Let $\mathcal{P}(\mathcal{D}_f)$ denote the *powerset* of the domain of an image f. Let $sat : \mathcal{P}(\mathcal{D}_f) \to \mathcal{P}(\mathcal{D}_f)$ be the operator for *filling holes* and $\mathcal{SAT}(f) = \{sat(C) : C \in \mathcal{L}(f) \cup \mathcal{U}(f)\}$ be the family of CCs of the upper and lower level sets without holes. The elements of $\mathcal{SAT}(f)$, called *shapes*, are nested by an inclusion relation and thus the pair $(\mathcal{SAT}(f), \subseteq)$, induces the tree of shapes [9], [11], [12], which leads to Prop. 2. It is worth noting that in this tree, the leaves coincide with regional extrema (minima and maxima) of the image.

Proposition 2. *The ordered set $(\mathcal{SAT}, \subseteq)$ is a tree.*

It was shown that it is possible to build a tree of shapes of an image by merging its max-tree and min-tree [9], [11], [13]. The algorithm performs as follows: initially, both the min-tree and the max-tree of the input image are computed, but during this computation, a pixel belonging to each hole is stored. After, for each CC of min-tree $C_{\mu,k} \in \mathcal{U}(f)$ that has

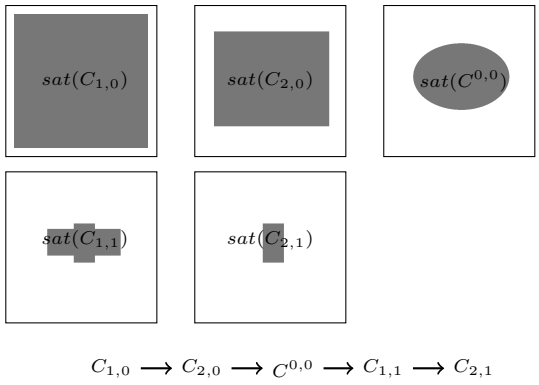

Figure 3. An example of tree of shapes of the input image shown in Fig. 2.

one or more holes, we seek in the max-tree, the CCs that fill the holes of $C_{\mu,k}$. After that, all the holes of the min-tree are filled, we fill the holes of the CCs of the max-tree with the CCs of min-tree [9].

The min-tree and max-tree are non-redundant representations and each contains all the pixels of the image domain [7]. Then, we can have duplicated pixels in the tree. Therefore, the last step of the algorithm consists of rebuilding a non-redundant tree of shapes by eliminating those duplicated pixels from its nodes. Fig. 3 shows the saturated CCs of the image f presented in Fig. 2 and its respective tree of shapes.

III. Connected filtering

As previously stated, the main advantage of using a tree structure to represent an image is that it can be used to perform filtering efficiently. But this filtering must be based on some criterion and this requires some information to be stored in the tree nodes.Many different kinds of information can be used, and we call them *attributes*. Formally, an attribute is a function $\kappa : \mathcal{P}(D_f) \rightarrow \mathbb{R}_+$ and it can represent node information such as area, volume, height, width, circularity. Filtering an image by using an attribute can be done by simply pruning tree nodes that satisfy a certain *criterion* (for example, nodes whose attribute value is lower than a given threshold).

Definition 2. *Let $\mathcal{T}_f$ be the tree of an image f, then a* pruning *of $\mathcal{T}_f$ is the operation of removing all nodes $\mathcal{N}$ in $\mathcal{T}_f$ that satisfy a certain criterion on $\mathcal{N}$. Furthermore, the pixels stored in $\mathcal{N}$ are taken to the first preserved ancestor starting from $\mathcal{N}$ to the root.*

Some attributes can be increasing, i.e., if $A, B \in \mathcal{P}(\mathcal{D}_f)$ and $A \subseteq B$, then $\kappa(A) \leq \kappa(B)$. If the chosen tree is the max-tree (resp. min-tree), then the resulting filter by increasing attributes is the operator called *attribute opening* (resp., *closing*) [7], whereas if it is the tree of shapes, filtering results in a filter by *self-dual attributes* or *grain filter* [9], since it gives symmetrical treatment to extrema regions.

Definition 3. *A* connected operator *maps an image f into an image g such as the following relation is valid for all pairs of neighboring pixels: $\forall (p, q)$ neighbors, $f(p) = f(q) \Rightarrow g(p) = g(q)$.*

A connected operator equivalently satisfies the following relationship: $\forall (p, q)$ neighbors, if $g(p) \neq g(q)$ then $f(p) \neq f(q)$. It is easy to check that the attribute opening is a connected operator: if two neighboring pixels (p, q) satisfy $f(p) = f(q)$, then we have p and q belonging to the same level set and therefore to the same CC. Then, these two pixels will be in the same node and they have the same gray level, i.e., $g(p) = g(q)$.

A. Levelings

Meyer [14], [15] extensively studied connected operators specializations. One of these specializations, known as *levelings*, are powerful simplifying filters that preserve contours and extend flat zones of an image.

Definition 4. *A* leveling *operator maps an image f into an image g such that the following relation is valid for all pairs of neighboring pixels, i.e., $\forall (p, q)$ neighbors:*

$$g(p) > g(q) \Rightarrow f(p) \geq g(p) \text{ and } g(q) \geq f(q).$$

Levelings are increasing operators, since $f(p) \geq g(p) > g(q) \geq f(q) \Rightarrow f(p) > f(q)$. Moreover, it is possible to show that if g is the result of filtering using increasing attributes from the max-tree, min-tree or tree of shapes built from an image f, then g is a leveling of f.

Theorem 1. *Let f be an image and consider the tree (max-tree, min-tree or tree of shapes) that represents f. Consider that the tree was filtered by an increasing attribute and this pruned tree was reconstructed into an image g. Then, g is a leveling of f.*

Proof: To prove that g is a leveling of f, we need to prove that the leveling definition holds for every pair of neighboring pixels within the domain of the image.

This property can be easily proved if the tree used is a component tree: assume, for instance, that a max-tree is being used. Let $\mathcal{C}^f(p)$ be the node of $\mathcal{T}_f$ that represents the smallest CC that contains the pixel p. By definition, $\mathcal{C}^f(p)$ is a CC of $\mathcal{X}_{f(p)}(f)$. Also, let $\mathcal{C}^g(p)$ be the node representing the smallest CC that contains p in $\mathcal{T}_g$, where $\mathcal{T}_g$ is the tree obtained by pruning $\mathcal{T}_f$. Since pruning the tree can only eliminate nodes, it is clear that $\mathcal{C}^f(p) \subseteq \mathcal{C}^g(p)$.

Consider a pair (p, q) of neighboring pixels. If $f(p) = f(q)$, then $\mathcal{C}^f(p) = \mathcal{C}^f(q)$, since their gray levels are equal and p and q are neighbors. Then, any pruning in the tree either would keep or eliminate $\mathcal{C}^f(p)$ from the reconstructed image but, in both cases, the nodes $\mathcal{C}^g(p)$ and $\mathcal{C}^g(q)$ will be the same node, implying $g(p) = g(q)$, which satisfies the leveling definition because the hypothesis $g(p) > g(q)$ is false.

Now, consider $f(p) \neq f(q)$. Without loss of generality, suppose $f(p) > f(q)$. Then, by definition, $\mathcal{C}^f(p) \subseteq \mathcal{X}_{f(p)}(f) \subset \mathcal{X}_{f(q)}(f)$. Consequently, $\mathcal{C}^f(p) \subseteq \mathcal{C}^f(q)$, since $\mathcal{C}^f(q)$ is a maximal set of $\mathcal{X}_{f(q)}(f)$ that contains the neighboring pixels p and q. If neither of these nodes is pruned, then we have $f(p) = g(p) > g(q) = f(q)$, which can be easily seen that satisfies the leveling definition with some algebric manipulations. If both nodes are pruned, then $\mathcal{C}^g(q)$ must contain both p and q,

since $\mathcal{C}^f(q) \subseteq \mathcal{C}^g(q)$ and $\mathcal{C}^f(q)$ already contained these pixels, implying $\mathcal{C}^g(p) = \mathcal{C}^g(q) \Rightarrow g(p) = g(q)$. The only other possibility is the case when only $\mathcal{C}^f(p)$ is pruned (if $\mathcal{C}^f(q)$ is pruned, $\mathcal{C}^f(p)$ will also be, since $\mathcal{C}^f(p) \subseteq \mathcal{C}^f(q)$). In this case, we will have that $\mathcal{C}^f(p) \subseteq \mathcal{C}^g(p) \subseteq \mathcal{C}^g(q) = \mathcal{C}^f(q)$. If $\mathcal{C}^g(p) = \mathcal{C}^g(q)$ then $g(p) = g(q)$ and there is nothing to prove. Otherwise, $\mathcal{C}^g(p) \subset \mathcal{C}^g(q) \Rightarrow f(p) \geq g(p) > g(q) = f(q)$, which also satisfies the leveling definition.

So far, it was shown that, for any case and for any pair of neighboring pixels, the leveling definition still holds. Similar arguments can be used to prove that proposition when a min-tree is used.

The complete proof for the case of tree of shapes is more difficult and then only some insights will be given. Let $\mathcal{S}^f(p)$ be the node of a tree of shapes $\mathcal{T}_f$ that represents the smallest CC that contains p and suppose two neighboring pixels (p, q). If $f(p) = f(q)$ then $\mathcal{S}^f(p) = \mathcal{S}^f(q)$, since their gray levels are equal and p and q are neighbors. This implies $g(p) = g(q)$, which satisfies the leveling definition because the condition $g(p) > g(q)$ is false.

Now, consider $f(p) \neq f(q)$. Then, we have two cases: either $\mathcal{S}^f(p) \subset \mathcal{S}^f(q)$ or $\mathcal{S}^f(p) \cap \mathcal{S}^f(q) = \emptyset$ (see Theorem 2.16 in [13]).

- If $\mathcal{S}^f(p) \subset \mathcal{S}^f(q)$ then the branch of nodes $(\mathcal{S}^f(q) = \mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_n = \mathcal{S}^f(p))$ linking $S^f(q)$ to $S^f(p)$ must satisfy one of the following properties:
  1) $f(\mathcal{S}_i) > f(\mathcal{S}_{i+1})$, $1 \leq i < n$, if $f(q) > f(p)$;
  2) $f(\mathcal{S}_i) < f(\mathcal{S}_{i+1})$, $1 \leq i < n$, if $f(q) < f(p)$.

  In the above property, $f(\mathcal{S}_i)$ will denote the gray level of the pixels of the node $\mathcal{S}_i$. If $f(q) = f(p)$, then it can be shown that p and q are not neighbors. Thanks to these properties, if the node $\mathcal{S}^f(q)$ is an ancestor of the node $\mathcal{S}^f(p)$, the gray level of the nodes between them follow a hierarchy similar to the min-tree (Prop. 1) or the max-tree (Prop. 2). Then, all the proofs used to prove that the pruning of component trees generates leveling are also valid.
- If $\mathcal{S}^f(p) \cap \mathcal{S}^f(q) = \emptyset$ then (p, q) are not in the same branch of the tree. In this case, there certainly exists a node $\mathcal{S}^f(r)$ that is a common ancestor of both $\mathcal{S}^f(p)$ and $\mathcal{S}^f(q)$ (in the worst case, this common ancestor is the root, the node representing the whole image). It can be shown that the branches linking $\mathcal{S}^f(r)$ to $\mathcal{S}^f(p)$ and $\mathcal{S}^f(r)$ to $\mathcal{S}^f(q)$ satisfy the hierarchy of gray levels presented in the above property. Furthermore, if one the branches satisfies Prop. 1, the other branch must satisfy Prop. 2. Using this fact, a relation between the gray levels of $\mathcal{S}^f(p)$, $\mathcal{S}^f(q)$ and $\mathcal{S}^f(r)$ can be obtained and it is possible to show that any pruning regarding pixels p and q still satisfy the leveling definition.

■

Levelings can be nested to create a space-scale decomposition of an image [16]. Suppose $\mathcal{T}_f$ is a tree that represents an image f and g is the result of a filtering by increasing attribute in $\mathcal{T}_f$ by a threshold λ (i.e., a pruning in $\mathcal{T}_f$). Denote by $\mathcal{T}_g$ this pruned version of $\mathcal{T}_f$. Likewise, consider the tree $\mathcal{T}_h$ pruned obtained by a filtering of an increasing attribute in $\mathcal{T}_g$ by a threshold $\lambda + 1$.

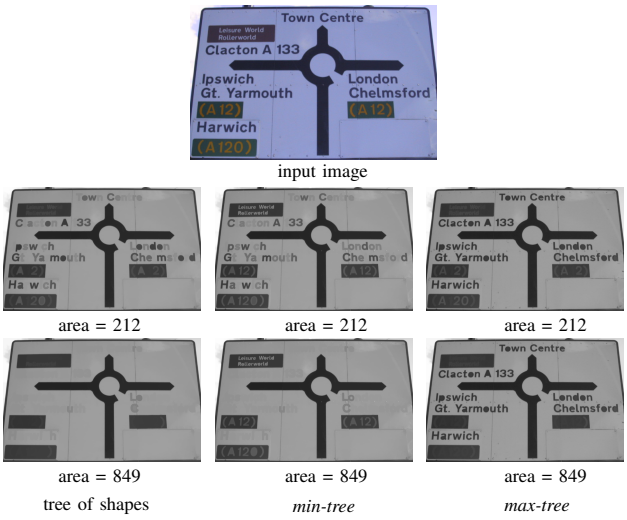

Figure 4. Simplifications of images by levelings.

Hence, by Theo. 1, we have h is a leveling of g and g is a leveling of f. Then, from Def. 4, we have that for all pairs of neighboring pixels (p, q), $h(p) > h(q) \Rightarrow g(p) \geq h(p) > h(q) \geq g(q) \Rightarrow f(p) \geq g(p) \geq h(p)$ and $h(q) \geq g(q) \geq f(q)$, showing that h is also a leveling of f. This shows that the tree generates a family of levelings that further simplifies the image f, thus constituting a morphological scale space with the follwing features [5], [16], [17].

- *Simplification*: from one scale to the next, a real simplification takes place.
- *Causality*: coarser scales can only be caused by what happened at finer scales.
- *Fidelity*: It does not create new structures at coarser scales, in other words, each regional minima (resp. maxima) of the leveling contains a regional minima (resp. maxima) of the reference image [14].

This leads us to the following proposition:

Proposition 3. *Let $\mathcal{T}_f$ be a tree (max-tree, min-tree or tree of shapes) that represents the image f and f_λ the filtered image by an increasing attribute of value $\lambda \geq 0$. Then, $(f_\lambda, f_{\lambda+1}, ..., f_{\lambda+n})$ is a space-scale of levelings such that $f_{\lambda+k}$ is a leveling of $f_{\lambda+l}$ for all $0 \leq l \leq k$.*

Fig. 4 show examples of simplifications obtained by levelings using min-tree, max-tree and tree of shapes.

IV. Contrast extraction using numerical residues

In this section, we present the ultimate attribute opening (resp. closing) [4], [8], [6], [18], [19] and the method of residues extraction proposed by Meyer [5]. They compose the background of the proposed method.

A. Ultimate attribute opening

The *ultimate attribute opening* $\theta(f) = (R_\theta(f), q_\theta(f))$ (resp., *closing*) of an image $f \in \mathcal{F}_\mathcal{D}$ is given by two operators $R_\theta : \mathcal{F}_\mathcal{D} \to \mathcal{F}_\mathcal{D}$ and $q_\theta : \mathcal{F}_\mathcal{D} \to \mathcal{F}_\mathcal{D}$ defined as follows:

$$R_\theta(f) = \sup\{r_\lambda(f) : r_\lambda(f) = \rho_\lambda(f) - \rho_{\lambda'}(f)\},$$

$$[q_\theta(f)](x) = \max\{\lambda + 1 : [r_\lambda(f)](x) = [R_\theta(f)](x) > 0\},$$

where ρ_λ and $\rho_{\lambda'}$ are, respectively, the consecutive attribute openings γ_λ and $\gamma_{\lambda+1}$ (resp., the attribute closings $\phi_{\lambda+1}$ and ϕ_λ) with attribute value λ. Note that $\gamma_{\lambda+1}(f) \leq \gamma_\lambda(f)$ (resp. $\phi_{\lambda+1}(f) \geq \phi_\lambda(f)$). For this reason, the residues given by $R_\theta(f)$ always have non-negative values.

An attribute opening γ_λ can be computed in a max-tree by removing nodes $C_{h,k}$ whose attribute $\kappa(C_{h,k}) < \lambda$. In fact, Fabrizio and Marcotegui [4] has proposed a fast implementation of the ultimate attribute opening using max-tree. As γ_λ is obtained by pruning nodes from the tree, the residue r_λ (i.e., $\gamma_\lambda - \gamma_{\lambda+1}$) can be calculated as the difference between the gray level of the removed nodes by pruning and the gray level of their respective parent nodes if their attribute values are different. If they are equal, both are removed by the same attribute opening and the residue is computed as the difference of graylevels of the removed node and its first ancestor node with a different attribute value, that is

$$r_\lambda(C_{h,k}) = \begin{cases} h - h' & \text{, if } \kappa(C_{h,k}) \neq \kappa(C_{h',k'}), \\ h - h' + & \\ \quad r_\lambda(C_{h',k'}) & \text{, otherwise.} \end{cases}$$

where $C_{h',k'}$ is the parent node (note that, $C_{h,k} \subset C_{h',k'}$ and $h' < h$).

This algorithm uses a top-down approach: the residue of the parent node is calculated; it is then propagated to the child nodes. Every child node compares its residue with the residue of the parent node and keeps the maximum of these two values in R_θ. At the same time, it also stores q_θ, the attribute value of $\kappa(C_{h,k})+1$ that generated the maximum residue. After this step, each child node becomes a parent node and the process is repeated.

B. Residues using a family of levelings

Meyer, in [5], proposed the extraction of residues of a family of levelings obtained by using families of anti-extensive erosions ε^λ and of extensive dilatations δ^λ of size λ as markers. More formally, given an image f, the family of levelings is obtained from: $l_\lambda(f) = lev(lev(l_{\lambda-1}, \varepsilon^\lambda(f)), \delta^\lambda(f))$, where $l_0 = f$ and $lev : \mathcal{F}_\mathcal{D} \times \mathcal{F}_\mathcal{D} \to \mathcal{F}_\mathcal{D}$ is the operator that, when applied to f and g (i.e., $lev(f, g)$), transforms the marker g into a leveling of f (for more details, see [5], [14], [15]).

The difference between two consecutives levelings produces a residue with positive $r^+_\lambda = [0 \vee (l_\lambda - l_{\lambda-1})]$ and negative $r^-_\lambda = [0 \vee (l_{\lambda-1} - l_\lambda)]$ parts, where $\vee$ denotes the supremum operation. Then, the maximum residues are extracted throughout the decomposition of the image, separating the positive R^+_θ and negative R^-_θ parts:

$$R^-_\theta(f) = \sup\{r^-_\lambda(f) : r^-_\lambda(f) = [0 \vee (l_{\lambda-1}(f) - l_\lambda(f))]\},$$

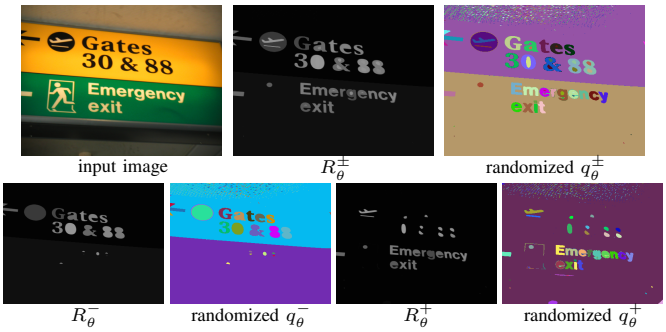

Figure 5. Example of ultimate attribute leveling.

$$R^+_\theta(f) = \sup\{r^+_\lambda(f) : r^+_\lambda(f) = [0 \vee (l_\lambda(f) - l_{\lambda-1}(f))]\}.$$

If N is the number of pixels ($N = |\mathcal{D}_f|$), an implementation for extraction of residues (referred here as the *naive algorithm*) obtained by the above definitions needs to calculate, for every λ, an erosion ε_λ (complexity $O(\lambda N)$); a dilation δ_λ (complexity $O(\lambda N)$); two lev (complexity $O(\lambda N)$) transforms; and positive and negative residues (complexity $O(N)$). Then, as λ is the maximum amount of pixels, the computation of these residues is, in the worst case, $O(N^2)$.

V. Ultimate attribute leveling

As original contribution of our work, we present in this section a residual operator, called *ultimate attribute leveling*. We also present an efficient algorithm for computing it based on the tree of shapes. This algorithm is a combination of the positive and negative residues extracted from levelings [5] with the recent results of ultimate attribute opening [4], [6], [8], [18], [19].

The family of levelings used by Meyer, to extract the residues, has as characteristic the simplification of the image by regional extrema. Thus, one can obtain similar simplifications using a family of levelings obtained by pruning the tree of shapes (see Prop. 3). Therefore, the filtered image is simplified by regional extrema, since the nodes of the tree represent regional extrema of the image. Moreover, we can select the CCs for extraction of residues by changing the type of increasing attribute from tree and still obtain the same characteristic.

Combining the ultimate attribute opening with the ideas of Meyer, we have the ultimate attribute leveling $R^\pm_\theta$ (see example in Fig. 5), where primitives are levelings obtained from the tree of shapes and so, following Meyer, separating positive and negative residues. More formally, the *ultimate attribute leveling* $R^\pm_\theta(f) = (R^\pm_\theta(f), q^\pm_\theta(f))$ of an image $f \in \mathcal{F}_\mathcal{D}$ is given by two operators $R^\pm_\theta : \mathcal{F}_\mathcal{D} \to \mathcal{F}_\mathcal{D}$ and $q^\pm_\theta : \mathcal{F}_\mathcal{D} \to \mathcal{F}_\mathcal{D}$ defined as follows:

$$[R^\pm_\theta(f)](x) = \max\{[R^-_\theta(f)](x), [R^+_\theta(f)](x)\},$$

$$[q^\pm_\theta(f)](x) = \begin{cases} [q^+_\theta(f)](x) & \text{, if } [R^+_\theta(f)](x) > [R^-_\theta(f)](x), \\ [q^-_\theta(f)](x) & \text{, otherwise,} \end{cases}$$

where

$$R^-_\theta(f) = \sup\{r^+_\lambda(f) : r^+_\lambda(f) = [0 \vee l_\lambda(f) - l_{\lambda+1}(f)]\},$$

$$R_\theta^+(f) = \sup\{r_\lambda^-(f) : r_\lambda^-(f) = [0 \vee l_{\lambda+1}(f) - l_\lambda(f)]\},$$
$$[q_\theta^+(f)](x) = \max\{\lambda + 1 : [r_\lambda^+(f)](x) = [R_\theta^+(f)](x) > 0\},$$
$$[q_\theta^-(f)](x) = \max\{\lambda + 1 : [r_\lambda^-(f)](x) = [R_\theta^-(f)](x) > 0\}.$$

Now, we present an efficient algorithm for ultimate leveling computation using tree of shapes. This algorithm is based on the algorithm presented by Fabrizio and Marcotegui [4] to compute the ultimate opening using max-tree and the concept of gradual transition shown in [19]. The efficiency of our algorithm is based on computing positive and negative residues using the tree of shapes; levelings properties; and recent results of the ultimate attribute opening.

Algorithm 1: Compute the ultimate attribute leveling.

Input: A tree of shapes $\mathcal{T}$; the maximum filter criterion *MAX* and value Δ of the gradual transition.
Output: Images: R_θ^-, R_θ^+, $R_\theta^\pm$, q_θ^+, q_θ^-, $q_\theta^\pm$.
Data: Auxiliar vectors: r^+, r^-, q_r^+, q_r^-.

for *i from 1 to number of nodes of* $\mathcal{T}$ **do**
  $r^+(i) = r^-(i) = q_r^+(i) = q_r^-(i) = -1$;
Let $C_{\lambda,0}$ be the root of $\mathcal{T}$
foreach *node* C_i *child of* $C_{\lambda,0}$ **do**
  ComputeUAL(r^+, r^-, q_r^+, q_r^-, C_i, λ, *MAX*, Δ);
foreach *pixel* $p \in D_f$ **do**
  Let k be the index of smallest CC (node) $C_{\lambda,k}$ that contains p.
  $R_\theta^-(p) = r^-(k)$; $R_\theta^+(p) = r^+(k)$;
  $q_\theta^+(p) = q_r^+(k)$; $q_\theta^-(p) = q_r^-(k)$;
  $R_\theta^\pm(p) = \max\{R_\theta^+(p), R_\theta^-(p)\}$;
  if $R_\theta^+(p) > R_\theta^-(p)$ **then**
    $q_\theta^\pm(p) = q_\theta^+(p)$;
  else
    $q_\theta^\pm(p) = q_\theta^-(p)$;

Alg. 2 (used by Alg. 1) needs as input a tree of shapes. This tree can be built by merging the max-tree and the min-tree with complexity $O(N\log(N))$ [9], [11] as described in Section II. Once the tree of shapes is built, we execute Alg. 1, where, firstly, vectors are created to store the maximum residues and their associated values. Then, for each child C_i of the root ($C_{\lambda,0}$) of the tree, Procedure *ComputeUAL* (Alg. 2) is called, where it initiates the tree computation starting from node C_i.

Since Procedure *ComputeUAL* visits each node of the tree of shapes exactly once and the number of nodes of the tree is at most the number of pixels in the image, *ComputeUAL* is executed $O(N)$ times. Afterwards, the algorithm generates images of the ultimate attribute leveling from the output of the Algorithm *ComputeUAL*, visiting each pixel exactly once, so the complexity of the Alg. 1 is $O(N)$. Therefore, to extract the ultimate attribute leveling, we build the tree of shapes with complexity $O(N\log(N))$ and compute the residues (Alg. 1) with complexity $O(N)$. Thus, the complexity of the proposed algorithm is $O(N\log(N)) + O(N) = O(N\log(N))$.

Algorithm 2: ComputeUAL(Vector r^+, Vector r^-, Vector q_r^+, Vector q_r^-, node $C_{\lambda,k}$, Integer prevLevel, Integer *MAX*, Integer Δ)

Output: Vectors: r^+, r^-, q_r^+, q_r^-.
Data: Auxiliar Integers: attr$^+$, attr$^-$, res$^+$, res$^-$, level.

Let $C_{\lambda',k'}$ be the parent node of $C_{\lambda,k}$.
if $|\kappa(C_{\lambda',k'}) - \kappa(C_{\lambda,k})| \leq \Delta$ **then**
  level = prevLevel;
else
  level = λ'
if $\kappa(C_{\lambda,k}) > MAX$ **then**
  res$^+$ = res$^-$ = attr$^+$ = attr$^-$ = 0;
else
  res$^+$ = max$\{0, \lambda$ - level$\}$;
  res$^-$ = max$\{0,$ level - $\lambda\}$;
  if $r^+(k') \geq res^+$ **then**
    attr$^+$ = $q_r^+(k')$;
  else
    attr$^+$ = $\kappa(C_{\lambda,k}) + 1$;
  if $r^-(k') \geq res^-$ **then**
    attr$^-$ = $q_r^-(k')$;
  else
    attr$^-$ = $\kappa(C_{\lambda,k}) + 1$;
$r^+(k)$ = max$\{$ $r^+(k')$, res$^+\}$;
$r^-(k)$ = max$\{$ $r^-(k')$, res$^-\}$;
$q_r^+(k)$ = attr$^+$; $q_r^-(k)$ = attr$^-$;
foreach *node* C_i *child of* $C_{\lambda,k}$ **do**
  ComputeUAL(r^+, r^-, rq^+, rq^-, C_i, level, *MAX*, Δ);

Fig. 6 shows that our algorithm is more efficient when the value of *MAX* (maximum value to calculate residues) increases. In this experiment, we used a machine with processor Intel i7 of 2.7GHz and 8GB RAM.

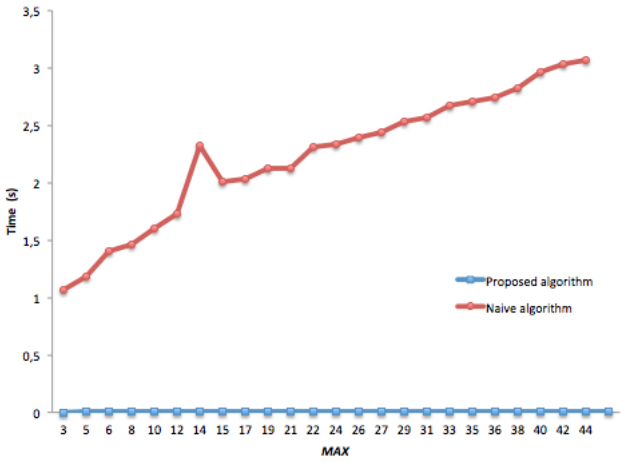

Figure 6. Execution time

VI. Extended ultimate attribute leveling by shape information

The extraction of numerical residues based on connected operators has been combined with a specific priori shape [8], [18]. In particular, the ultimate attribute opening, would be later, extended by Hernández and Marcotegui in [8], [18], where the residues of r_λ is combined with a similarity function based on the characteristics of CCs in images. Based on the ideas of Hernández and Marcotegui, we propose the ultimate attribute leveling with shape information.

A. Shape information

In this section, we define our shape similarity measure in order to combine prior shape knowledge with the ultimate attribute leveling operator. The shape definition has been widely studied in the literature [20]. In our context, given an image f, we are interested in comparing two CCs (shapes) $C_{i,k}, C_{j,k'} \in \mathcal{SAT}(f)$ via a similarity function ψ defined as follows.

Definition 5. *A similarity measure between two shapes, $C_{i,k}$ and $C_{j,k'}$ in $\mathcal{SAT}(f)$, is defined as a function $\psi : \mathcal{SAT}(f) \times \mathcal{SAT}(f) \to [0,1] \in \mathbb{R}$ verifying the following conditions [8], [20]:*

- *Identity:* $\psi(C_{i,k}, C_{i,k}) = 1$.
- *Uniqueness:* $\psi(C_{i,k}, C_{j,k'}) = 1$ *implies* $C_{i,k} = C_{j,k'}$.
- *Symmetry:* $\psi(C_{i,k}, C_{j,k'}) = \psi(C_{j,k'}, C_{i,k})$.

B. Ultimate attribute leveling with shape information

Following Hernández and Marcotegui in [8], we propose to consider a shape factor function $\psi(C_{i,k}, C_{ref})$ to a reference shape C_{ref} within the residue computation. As C_{ref} is fixed, $\psi(C_{i,k}, C_{ref})$ is denoted by $\psi(C_{i,k})$.

As the residue of a CC $C_{i,k} \in \mathcal{SAT}(f)$ is computed by $r_\lambda(C_{i,k})$ then the residue with shape information can be obtained by $r^\Omega_\lambda(C_{i,k}) = \psi(C_{i,k}) \cdot r_\lambda(C_{i,k})$. Now, we define the ultimate attribute leveling with shape information, as follows:

$$[R^{\Omega\pm}_\theta(f)](x) = \max\{[R^-_\theta(f)](x), [R^+_\theta(f)](x)\},$$

$$[q^{\Omega\pm}_\theta(f)](x) = \begin{cases} [q^+_\theta(f)](x) & \text{,if } [R^+_\theta(f)](x) > [R^-_\theta(f)](x), \\ [q^-_\theta(f)](x) & \text{,otherwise,} \end{cases}$$

where

$$R^-_\theta(f) = \sup\{r^{\Omega+}_\lambda(f) : r^{\Omega+}_\lambda(f) = \psi(C_{\lambda,k}) \cdot r^+_\lambda(f)\},$$

$$R^+_\theta(f) = \sup\{r^{\Omega-}_\lambda(f) : r^{\Omega-}_\lambda(f) = \psi(C_{\lambda,k}) \cdot r^-_\lambda(f)\},$$

$$[q^+_\theta(f)](x) = \max\{\lambda + 1 : [r^{\Omega+}_\lambda(f)](x) = [R^+_\theta(f)](x) > 0\},$$

$$[q^-_\theta(f)](x) = \max\{\lambda + 1 : [r^{\Omega-}_\lambda(f)](x) = [R^-_\theta(f)](x) > 0\}.$$

C. Example of shape information

In order to compare two shapes, we follow [8] to define a similarity function via shape descriptors. Thus, we use the simplest shape descriptors: geometric features (height, width, etc) and their relations (fill ratio, circularity, moments, etc), e.g.:

$$\psi_{circle}(C_{i,k}) = 4\pi \frac{Area(C_{i,k})}{Perimeter(C_{i,k})^2},$$

or

$$\psi_{rectangle}(C_{i,k}) = \frac{Area(C^{Rot}_{i,k})}{Width(C^{Rot}_{i,k}) \times Height(C^{Rot}_{i,k})},$$

where $C^{Rot}_{i,k}$ is a rotated version of $C_{i,k}$ in the direction of its major axis.

Furthermore, we can use heuristics of a given problem, to determine a shape factor function. Such as the heuristics to the text localization problem [21]:

$$\psi_{text}(C_{i,k}) = \psi_{area}(C_{i,k})\psi_{hole}(C_{i,k})\psi_{rect}(C_{i,k})\psi_{rate}(C_{i,k})\psi_{color}(C_{i,k}),$$

where:

$$\psi_{area}(C_{i,k}) = \begin{cases} 1 & \text{, if } 50 \le Area(C_{i,k}) \le Area(f/2), \\ 0 & \text{, otherwise,} \end{cases}$$

$$\psi_{hole}(C_{i,k}) = \begin{cases} 1 & \text{, if } Hole(C_{i,k}) \le 3, \\ 0 & \text{, otherwise,} \end{cases}$$

$$\psi_{rect}(C_{i,k}) = \begin{cases} 1 & \text{, if } 0.35 \le \psi_{rectangle}(C_{i,k}) \le 0.9, \\ 0 & \text{, otherwise,} \end{cases}$$

$$\psi_{rate}(C_{i,k}) = \begin{cases} 1 & \text{, if } \frac{\max\{Width(C_{i,k}), Height(C_{i,k})\}}{\min\{Width(C_{i,k}), Height(C_{i,k})\}} \le 4, \\ 0 & \text{, otherwise,} \end{cases}$$

$$\psi_{color}(C_{i,k}) = \begin{cases} 1 & \text{, if } Var(C_{i,k}) \le 15, \\ 0 & \text{, otherwise,} \end{cases}$$

where $Hole(C_{i,k})$ is the number of holes in $C_{i,k}$; $Var(C_{i,k})$ is the variance of the gray levels of pixels of $C_{i,k}$ (same type) in the input image. Of course, one can use a classifier to estimate the thresholds of ψ_{text} (as was done by [21]).

Another possibility is to perform matching of complex shapes using robust methods.

VII. Application Examples

Extraction contrast, through numerical residues, has been used successfully in various applications in a preprocessing step such as text location [6], [21] , segmentation of building facades [8], and restoration of historical documents [5].

A. Text localization

Contrast extraction using numerical residues have already been used in text localization problem [6], [18], [21]. Therefore, we selected some images with the presence of text to apply the ultimate attribute opening and closing, ultimate attribute leveling and ultimate attribute leveling with shape information. In Fig. 7, it can be observed that residues extracted by ultimate attribute leveling includes the residues of ultimate attribute opening and closing. Whereas the residues of ultimate attribute leveling with shape information ψ_{text} do not contain residues of the region without the presence of text.

Figure 7. Example extraction of residues applied to text location using height (bounding box) attribute.

VIII. Conclusion

The first contribuition of this paper is the introduction of a novel type of residual operators called ultimate attribute leveling that can be used for extracting contrast in gray level images. Generally speaking, ultimate attribute leveling was inspired from the definition of ultimate attribute opening and the residues extracted from families of levelings.

Another contribuition of this work is an efficient algorithm for ultimate attribute leveling computation based on tree of shapes. In fact, the time complexity of this is $O(N \log N)$, where N denotes the number of pixels of the input image. Experimental results (see Fig. 6) show that our algorithm is much faster than the naive algorithm.

Our algorithm for computating ultimate attribute leveling is based on Theo. 1, which states that levelings can be obtained by pruning nodes from tree of shapes. So far, it was known that levelings can be obtained from component trees. So, in this sense, Theo. 1 is also an original result of this work.

Finally, we propose the use of ultimate attribute leveling with shape information, which allows us to extract contrast using a priori knowledge of an application. In fact, contrast extraction by means of numerical residues has been used successfully in various applications, such as text location, segmentation of building facades and binarization of historical documents. Experimental results applied to text location show the potentiality of using ultimate attribute leveling with shape information.

For future research, we plan to study theoretical results of ultimate attribute leveling.

References

[1] H. J. A. M. Heijmans, *Morphological Image Operators*. Boston: Academic Press, 1994.

[2] J. Serra, *Image Analysis and Mathematical Morphology*. London: Academic Press, 1988, vol. 2: Theoretical Advances.

[3] S. Beucher, "Numerical residues," *Image Vision Comput.*, vol. 25, no. 4, pp. 405–415, 2007.

[4] J. Fabrizio and B. Marcotegui, "Fast implementation of the ultimate opening," in *Proceedings of the 9th International Symposium on Mathematical Morphology*, ser. ISMM '09, 2009, pp. 272–281.

[5] F. Meyer, "Levelings and flat zone morphology," *Pattern Recognition, International Conference on*, vol. 0, pp. 1570–1573, 2010.

[6] T. Retornaz and B. Marcotegui, "Scene text localization based on the ultimate opening," in *Proceedings of the 8th International Symposium on Mathematical Morphology and its Applications to Image and Signal Processing*, ser. ISMM '08, vol. 1, 2007, Image processing, pp. 177–188.

[7] P. Salembier, A. Oliveras, A. O. Member, and L. Garrido, "Anti-extensive connected operators for image and sequence processing," *IEEE Transactions on Image Processing*, 1998.

[8] J. Hernandez and B. Marcotegui, "Shape ultimate attribute opening," *Image and Vision Computing*, vol. 29, no. 8, pp. 533 – 545, 2011.

[9] P. Monasse and F. Guichard, "Fast computation of a contrast-invariant image representation," *IEEE Transactions on Image Processing*, vol. 9, no. 5, pp. 860–872, 2000.

[10] P. Salembier and L. Garrido, "Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval," *Image Processing, IEEE Transactions on*, vol. 9, no. 4, pp. 561–576, 2000.

[11] V. Caselles, E. Meinhardt, and P. Monasse, "Constructing the tree of shapes of an image by fusion of the trees of connected components of upper and lower level sets," *Positivity*, vol. 12, no. 1, pp. 55–73, 2008.

[12] G. Thierry, E. Carlinet, S. Crozet, and L. Najman, "A quasi-linear algorithm to compute the tree of shapes of nd images," in *Proceedings of the 11th International Symposium on Mathematical Morphology*, ser. ISMM '13, Uppsala, Sweden, 2013, pp. 98–110.

[13] V. Caselles and P. Monasse, *Geometric Description of Topographic Maps and Applications to Image Processing*, ser. Lecture Notes in Mathematics 1984, Book. Springer Verlag, 2010.

[14] F. Meyer, "From connected operators to levelings," in *Proceedings of the 4th International Symposium on Mathematical Morphology and its Applications to Image and Signal Processing*, 1998, pp. 191–198.

[15] ——, "The levelings," in *Proceedings of the 4th International Symposium on Mathematical Morphology and its Applications to Image and Signal Processing*, ser. ISMM '98, 1998, pp. 199–206.

[16] F. Meyer and P. Maragos, "Morphological scale-space representation with levelings," in *in Scale-Space?99, ser. Lecture Notes in Computer Science*. Springer-Verlag, 1999, pp. 187–198.

[17] P. Monasse and F. Guichard, "Scale-space from a level lines tree," *Journal of Visual Communication and Image Representation*, 2000.

[18] J. Hernandez and B. Marcotegui, "Ultimate attribute opening segmentation with shape information," in *Mathematical Morphology and Its Application to Signal and Image Processing*, 2009.

[19] B. Marcotegui, J. Hernandez, and T. Retornaz, "Ultimate opening and gradual transitions," in *Mathematical Morphology and Its Applications to Image and Signal Processing*, 2011, vol. 6671, pp. 166–177.

[20] R. C. Veltkamp and M. Hagedoorn, "Shape similarity measures, properties, and constructions," in *In Advances in Visual Information Systems, 4th International Conference*. Springer, 2000, pp. 467–476.

[21] W. A. L. Alves and R. F. Hashimoto, "Text regions extracted from scene images by ultimate attribute opening and decision tree classification," in *Proceedings of the 23rd SIBGRAPI Conference on Graphics, Patterns and Images*, 2010, pp. 360–367.